Amendments to the CLAIMS:

Without prejudice, this listing of the claims replaces all prior versions and listings of the claims in the present application:

LISTING OF CLAIMS:

The current status of the pending claims is listed below:

---

1. (Previously Presented) In a computer system, a method, performed at a manager, of distributing call flow events among a plurality of threads, each thread having an associated call flow event queue in which call flow events are queued, the method comprising:

B1

A. determining a call flow workload level for each of the plurality of threads;

B. determining that a first of the plurality of threads is inefficiently handling its assigned call flow workload; and

C. reassigning a call flow event from the call flow event queue associated with the first thread to the call flow event queue associated with a second of the plurality of threads.

2. (Previously Presented) The method according to claim 1 further comprising the step:

D. processing the call flow events associated with each of the plurality of threads.

3. (Currently Amended) The method according to claim 1 wherein step C further comprises:

C.1 removing a call flow event from the call flow event queue associated ~~within~~ with the first thread; and

C.2 placing the removed call flow event in the call flow event queue associated with the second thread.

4. (Previously Presented) The method according to claim 1 wherein step C further comprises:

C.1 selecting the second thread in accordance with the number of call flow events in the call flow event queue associated with the second thread.

5. (Previously Presented) The method according to claim 1 wherein step C further comprises:

C.1 allocating the call flow events to a thread within the computer system with the least call flow load.

6. (Previously Presented) The method according to claim 1 wherein step B further comprises:

B.1 determining whether the number of call flow events in the call flow event queue associated with a thread has exceeded a predetermined criteria.

7. (Previously Presented) The method according to claim 1, wherein step A comprises:

A.1 assigning call flow events among the call flow queues associated with the respective plurality of threads in the system.

8. (Currently Amended) A computer program product for use with a computer system, the computer system operatively coupled to a computer network and capable of communicating with one or more processes over the network, the computer program product comprising a computer usable medium having program code embodied in the medium, the program code being operable at a manager and comprising:

(A) program code configured to determine a call flow workload level for each of ~~the~~ a plurality of threads;

(B) program code configured to determine that a first of the plurality of threads is inefficiently handling its assigned call flow workload; and

(C) program code configured to reassign a call flow event from the call flow event queue associated with the first thread to the call flow event queue associated with a second of the plurality of threads.

9. (Previously Presented) The computer program product of claim 8, further comprising:

(D) program code configured to process the call flow events within each of the plurality of threads.

10. (Previously Presented) The computer program product according to claim 8 further comprising:

(C.1) program code configured to remove a call flow event from the call flow event queue associated within the first thread; and

(C.2) program code configured to place the removed call flow event in the call flow event queue associated with the second thread.

B1

11. (Previously Presented) The computer program product according to claim 8 further comprising:

(C.1) program code configured to select the second thread in accordance with the number of call flow events in the call flow event queue associated with the second thread.

12. (Previously Presented) The computer program product according to claim 8 further comprising:

(C.1) program code configured to allocate the call flow events to a thread within the computer system with the least call flow load.

13. (Previously Presented) The computer program product according to claim 8 further comprising:

(B.1) program code configured to determine whether the number of call flow events in the call flow event queue associated with a thread has exceeded a predetermined criteria.

14. (Previously Presented) The computer program product according to claim 8, further comprising:

(A.1) program code configured to assign call flow events among the call flow event queues associated with the respective plurality of threads in the system.

15. (Currently Amended) In a computer system, an apparatus for distributing call flow events among a plurality of threads, each thread having an associated call flow event queue in which call flow events are queued, the apparatus comprising:

a processor including:

a call flow engine configured to execute call flow events associated with one of the threads;

a call flow manager configured to distribute a plurality of call flow events among a plurality of threads used for managing the processing of a plurality of call flows, the call flow manager optimizing the processing of the call flows by determining which of the plurality of threads are operating inefficiently and reassigning a portion of the call flow events assigned to the inefficient thread to other of the plurality of threads having excess call flow processing capacity.

B1

16. (Previously Presented) The apparatus of claim 15 wherein the call flow manager continues to reassign call flow events until a balanced call flow event processing level is attained among the plurality of threads.

17. (Previously Presented) The method according to claim 1, further comprising:

D. determining whether a call flow balance has been achieved among the plurality of threads;

E. processing the call flow events associated with each of the plurality of threads.

18. (Previously Presented) The computer program product according to claim 8, further comprising:

(D) program code configured to determine whether a call flow balance has been achieved among the plurality of threads;

(E) program code configured to process the call flow events associated with each of the plurality of threads

19. (Previously Presented) The apparatus according to claim 15, wherein the call flow manager determines which of the plurality of threads are operating inefficiently by determining whether any of the threads has exceeded its maximum call flow capacity.